МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
Национальный университет кораблестроения
имени адмирала Макарова

Д.В. Костенко

ПРОЕКТИРОВАНИЕ КОМБИНАЦИОННЫХ УСТРОЙСТВ

Методические указания

Рекомендовано Методическим советом НУК

Николаев 2006

УДК 621.38 (076)

Костенко Д.В. Проектирование комбинационных устройств: Метод. указания. – Николаев: НУК, 2006. – 24 с.

Кафедра электрооборудования судов

Методические указания содержат основные сведения из алгебры логики, необходимые для анализа комбинационных устройств, способы минимизации и методики синтеза таких устройств на различной элементной базе. Все этапы минимизации и синтеза сопровождаются практическими примерами.

Предназначены для электротехнических специальностей.

Рецензент д–р техн. наук, проф. В.С. Блинцов.

ВВЕДЕНИЕ

В большинстве учебников по электронике и схемотехнике традиционно основное внимание уделяется устройству логических элементов, что, по мнению автора, является второстепенным вопросом. Далее рассматриваются функциональные особенности логических элементов и устройств средней степени интеграции, а методы минимизации и синтеза или отсутствуют вовсе, или приводятся самые простые, например минимизация с помощью карты Карно. Такой подход оправдан для электронных специальностей, где дискретная математика изучается как отдельная дисциплина.

Данные методические указания призваны восполнить указанный пробел для неэлектронных специальностей. Все изложенные вопросы снабжены примерами.

АЛГЕБРА ЛОГИКИ

Логические сообщения, операции, функции

Элементной базой вычислительной техники является цифровая электроника. Ее отличает то, что она оперирует не с физическими характеристиками сигналов, а с их логическим значением – истина или ложь, 1 или 0. Даже весьма сложные цифровые схемы строятся на принципе многократного повторения относительно простых базовых логических схем. Связи между этими схемами формируются на основе чисто формальных методов. Инструментом такого построения служит алгебра логики (булева алгебра), которая, в свою очередь, является математической интерпретацией формальной логики. Формальная логика оперирует *логическими сообщениями*, т. е. такими сообщениями, истинность или ложность которых может быть определена однозначно. Например: "прибор включен", "напряжение на входе меньше порога срабатывания". Логические сообщения формальной логики в булевой алгебре заменяются математическими эквивалентами – *логическими функциями*. Логическая функция может принимать только два значения – 1 (истина) или 0 (ложь).

При описании работы устройств важны не только логические функции, но и взаимосвязи между ними. Для математического описания связей между логическими функциями используют *логические операции*, которые затем при физической реализации заменяются *логическими элементами*. Логические элементы – готовые электронные устройства, которые выпускаются в виде микросхем и используются для построения цифровых устройств.

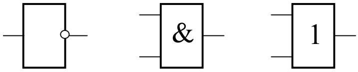
Существуют три основные логические операции:

операция НЕ (инверсия, отрицание): $y = \bar{x}$. Функция y принимает значение 1, если $x = 0$, и наоборот;

операция И (логическое умножение, конъюнкция): $y = x_1 x_2 = x_1 \& x_2 = x_1 \cap x_2$. Функция y принимает значение 1 в том и только в том случае, если обе функции x_1 и x_2 равны 1;

операция ИЛИ (логическое сложение, дизъюнкция): $y = x_1 + x_2 = x_1 \cup x_2$. Функция y принимает значение 1, если хотя бы одна из функций x_1 или x_2 равна 1.

Логические элементы, выполняющие указанные операции, имеют следующие обозначения:



Формы представления логических функций

Логические функции могут быть представлены различными формами:

в виде словесного описания. Например: Двигатель можно включить, если нажата кнопка "Пуск" И НЕ нажата кнопка "Стоп" И НЕ сработала защита от перегрузки;

в виде формулы. Для приведенного выше словесного описания формула будет иметь вид $d = a\bar{b}\,\bar{c}$;

в виде таблицы истинности. Таблица истинности должна содержать все возможные комбинации значений аргументов и соответствующие значения функции. Комбинации аргументов должны быть расположены по строкам таблицы так, чтобы полученные многобитные числа шли в порядке возрастания;

a	b	c	d
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

в виде перечисления порядковых номеров комбинаций аргументов, соответствующих истинному значению функции. В данном примере такая комбинация одна: $d = \sum(4)$ или $d = \bigcup(4)$.

Основы алгебры логики

Алгебра логики позволяет математически записывать логические сообщения и связи между ними, что необходимо для определения порядка и принципа работы устройств; реализовывать логические уравнения в виде схем, т. е. переходить от аналитического описания процесса к его схемной реализации в виде *логического автомата*; производить реализацию логических автоматов в оптимальном виде.

Логические операции между логическими функциями выполняются по следующим правилам:

$$x \cdot 1 = 1; \qquad x + 0 = x;$$
$$x \cdot 0 = 0; \qquad x + 1 = 1;$$
$$\overline{0} = 1; \qquad \overline{1} = 1.$$

По аналогии с обычной алгеброй в алгебре логики существует ряд теорем, которые называют основными тождествами алгебры логики.

Коммутативный закон: $ab = ba$, $a + b = b + a$.

Ассоциативный закон: $a(bc) = (ab)c$, $a + (b + c) = (a + b) + c$.

Дистрибутивный закон: $a(b + c) = ab + ac$, $a + bc = (a + b)(a + c)$.

Правило склеивания $(a + b)(a + \overline{b}) = a$, $ab + a\overline{b} = a$.

Правило поглощения $a(a + b) = a$, $a + ab = a$.

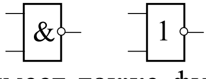
Правило повторения $aa = a$, $a + a = a$.

Правило отрицания: $a\overline{a} = 0$, $a + \overline{a} = 1$.

Правило двойного отрицания: $\overline{\overline{a}} = a$.

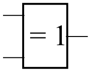
Теорема де Моргана: $\overline{ab} = \overline{a} + \overline{b}$, $\overline{a + b} = \overline{a} \cdot \overline{b}$.

В алгебре логики допускается подстановка одних функций вместо аргументов в другие функции (*суперпозиция функций*). Система логических функций, с помощью которых путем суперпозиции можно представить любую сколь угодно сложную функцию, называется *функционально полной*. Основные логические функции (НЕ, И, ИЛИ) образуют такую систему. Существуют и другие функционально полные системы, в том числе содержащие меньшее число функций. Функционально полные системы, состоящие только из одной функции, образуют функции И–НЕ (штрих Шеффера) и ИЛИ–НЕ (стрелка Пирса). Логические элементы, реализующие указанные функции, имеют следующие обозначения:



Важное значение имеет также функция $y = x_1 \oplus x_2$ – "исключающее ИЛИ" (сумма по модулю 2, отрицание равнозначности), которая имеет следующие таблицу истинности и схемное обозначение:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



Канонические формы представления логических функций

Одной и той же логической функции могут соответствовать различные суперпозиции функций функционально полной системы. Возникает задача получения такой формы представления функции, при которой каждой функции соответствует одна и только одна суперпозиция. Такие формы записи называются *каноническими*. Произвольную логическую функцию $f(X)$, где $X = (x_1, x_2, ..., x_n)$, можно представить в виде

$$f(X) = \bigcup_{A} x_1^{a_1} x_2^{a_2} \dots x_n^{a_n} f(A),$$

где $\bigcup_{A}$ – дизъюнкция по всем наборам;

$$A = (a_1, a_2, ..., a_n),\ \ a_i \in \{0,1\};$$

$$x_i^{a_i} = \begin{cases} \overline{x}, a_i = 0; \\ x, a_i = 1. \end{cases}$$

Используя свойства конъюнкции, состоящие в том, что $0x = 0$, $1x = x$, получим $f(X) = \bigcup_{A^*} x_1^{a_1} x_2^{a_2} \dots x_n^{a_n}$, где символ A^* означает, что конъюнкции берут только по тем наборам A, на которых $f(A) = 1$. Такая форма называется *совершенной дизъюнктивной нормальной формой* (СДНФ) или формой И–ИЛИ. Иногда целесообразно использовать другие канонические формы, которые можно получить из СДНФ с помощью правила двойного отрицания и теоремы де Моргана:

$$f(X) = \overline{\overline{f(X)}} = \overline{\overline{\bigcup_{A^*} x_1^{a_1} x_2^{a_2} \dots x_n^{a_n}}} = \overline{\bigcap_{A^*} \overline{x_1^{a_1} x_2^{a_2} \dots x_n^{a_n}}}.$$

Полученная форма называется канонической формой И–НЕ/И–НЕ. Применив к полученному выражению теорему де Моргана еще раз, получим каноническую форму ИЛИ–И–НЕ:

$$\overline{f(X) = \bigcap_{A^*} \overline{x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n}}} = \overline{\bigcap_{A^*} (\overline{x_1^{a_1}} \cup \overline{x_2^{a_2}} \cup \ldots \cup \overline{x_n^{a_n}})}.$$

Аналогично может быть получена каноническая форма ИЛИ–НЕ / ИЛИ:

$$f(X) = \overline{\bigcap_{A^*} (\overline{x_1^{a_1}} \cup \overline{x_2^{a_2}} \cup \ldots \cup \overline{x_n^{a_n}})} = \bigcup_{A^*} \overline{(\overline{x_1^{a_1}} \cup \overline{x_2^{a_2}} \cup \ldots \cup \overline{x_n^{a_n}})}.$$

Еще четыре канонические формы можно получить, записав отрицание функции $f(X)$:

$$\overline{f(X)} = \bigcup_{A^{**}} x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n},$$

где символ A^{**} означает дизъюнкцию по всем наборам аргументов, при которых функция принимает значение $f(X)=0$. Используя аналогичные преобразования, можно получить канонические формы И–ИЛИ–НЕ, И–НЕ / И, ИЛИ–НЕ, ИЛИ–НЕ / ИЛИ–НЕ. Наибольшее значение имеет форма ИЛИ–И, которая еще называется *совершенной конъюнктивной нормальной формой*:

$$f(X) = \overline{\bigcup_{A^{**}} x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n}} = \bigcap_{A^{**}} \overline{x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n}} = \bigcap_{A^{**}} (\overline{x_1^{a_1}} \cup \overline{x_2^{a_2}} \cup \ldots \cup \overline{x_n^{a_n}}).$$

При построении канонических форм используют простейшие выражения, представляющие собой конъюнкции или дизъюнкции нескольких попарно различных аргументов, взятых с инверсиями или без них. Эти выражения называют соответственно элементарным произведением (конъюнкцией, *термом*) и элементарной дизъюнкцией (*дизъюнктивным термом*). Элементарное произведение, содержащее все переменные и равное 1 только при одном наборе аргументов, называется *конститутиентой единицы или минитермом*. Элементарная дизъюнкция, содержащая все переменные и равная 0 только при одном наборе аргументов, называется *конститутиентой нуля* или *макстермом*. Использование свойства дистрибутивности функций, входящих в функционально полную систему, позволяет получать так называемые *скобочные* формы логических функций, у которых общие части нескольких элементарных произведений, или дизъюнкций, вынесены за скобки.

Минимизация логических функций

Как было сказано выше, логическую функцию можно представить различными суперпозициями функций функционально полной системы. Для физической реализации интерес представляет суперпозиция, требующая минимума аппаратных затрат. Количество оборудования, требуемого для реализации устройства, называется его ценой. Процесс нахождения представления функции с минимальной ценой называется *минимизацией*. Наиболее объективный метод определения цены – это подсчет суммарного числа входов всех элементов в устройстве (*цена по Квайну*).

Функцию $\varphi(X)$ называют импликантой функции $f(X)$, если на любом наборе аргументов выполняется неравенство $f(X) \geq \varphi(X)$. Если функции $\varphi(X)$ и $\psi(X)$ – импликанты функции $f(X)$, то дизъюнкция $\varphi(X)$ и $\psi(X)$ также является импликантой функции $f(X)$. Простой импликантой функции $f(X)$ называют такое элементарное произведение, которое является импликантой функции $f(X)$, но никакая его собственная часть не является импликантой функции $f(X)$. Под собственной частью произведения $C = x_1^{a_1} x_2^{a_2} \ldots$ $\ldots x_n^{a_n}$ понимают такое произведение B, которое можно получить из C путем вычеркивания одной или нескольких переменных $x_i^{a_i}$.

Любая функция имеет конечное множество простых импликант, число которых не больше числа минитермов в СДНФ. Дизъюнкция всех простых импликант функции $f(X)$ равна этой функции и называется *сокращенной дизъюнктивной нормальной формой* (ДНФ). Получение ДНФ возможно *методом группировки*. Элементарные конъюнкции K_i и K_j в СДНФ группируются в пары так, чтобы после вынесения за скобку общего множителя K подформула $K_i \cup K_j$ имела вид $K(\bar{x} \cup x)$ или $K(1 \cup x)$. Далее эти выражения заменяются эквивалентными им выражениями K. Например:

$$\bar{x}_1 x_2 \bar{x}_3 \cup \bar{x}_1 x_2 x_3 \cup x_1 x_2 \bar{x}_3 \cup x_1 x_2 x_3 = \bar{x}_1 x_2 (\bar{x}_3 \cup x_3) \cup$$
$$\cup\, x_1 x_2 (\bar{x}_3 \cup x_3) = \bar{x}_1 x_2 \cup x_1 x_2 = x_2 (\bar{x}_1 \cup x_1) = x_2.$$

Функция $\varphi(X)$ накрывается функцией $f(X)$, если на любом наборе аргументов A $f(A) \geq \varphi(A)$. В общем случае сокращенная ДНФ не будет минимальной, так как некоторые простые импликанты могут накрываться дизъюнкцией других членов и их можно исключить. *Минимальная* ДНФ (МДНФ) должна состоять исключительно из простых импликант. Нахождение МДНФ сводится к выделению некоторого числа простых импликант данной функции из всего их множества и может осуществляться различными методами.

Метод Блейка основан на применении правил обобщенного склеивания $xK_1 \cup \bar{x}K_2 = xK_1 \cup \bar{x}K_2 \cup K_1K_2$ и поглощения. На первом этапе выполняются все возможные правила обобщенного склеивания. Первый терм сравнивается со вторым, третьим и так далее. Затем второй сравнивается с третьим, четвертым и так далее, в том числе с вновь полученными во всех предыдущих склеиваниях. На этом этапе формула усложняется. На втором этапе производятся все возможные операции поглощения. Для этого в преобразованной после операции склеивания формуле находятся термы наименьшего ранга и все термы в ДНФ, содержащие эти термы в качестве сомножителей, вычеркиваются.

Метод Квайна. Пусть функция задана в СДНФ. На первом этапе методом группировки отыскиваются все простые импликанты, т. е. формируется сокращенная ДНФ. Далее составляется импликантная таблица.

Простые импликанты	3	4	5	7	9	*B*	*C*	*E*	*F*
$x_3 x_4$	x			x		x			x
$\bar{x}_1 x_2 \bar{x}_3$		x	x						
$x_2 \bar{x}_3 \bar{x}_4$		x					x		
$\bar{x}_1 x_2 x_4$			x	x					
$x_1 \bar{x}_2 x_4$					x	x			
$x_1 x_2 \bar{x}_4$							x	x	
$x_1 x_2 x_3$								x	x

Столбцы этой таблицы отмечаются минитермами, строки – простыми импликантами. Пусть задана функция четырех переменных $f(X) = \bigcup(3,4,5,7,9,B,C,E,F)$.

Второй этап – расстановка меток. Ставятся метки там, где минитермы покрываются первичными импликантами.

Третий этап – нахождение существенных импликант. Если в столбце всего одна метка, то соответствующая импликанта называется *существенной*. Вычеркиваются все столбцы, которые покрывает эта импликанта.

Четвертый этап – вычеркивание лишних столбцов. Если в таблице есть два столбца с метками в одинаковых строках, то один из них вычеркивается (так как покрытие одного обеспечивает и покрытие другого).

Пятый этап – вычеркивание лишних первичных импликант. Если после четвертого этапа в таблице есть строки без единой метки, то они вычеркиваются.

Шестой этап – выбор минимального покрытия. В данном примере $f(X) = x_3x_4 \cup \bar{x}_1x_2\bar{x}_3 \cup x_1\bar{x}_2x_4 \cup x_1x_2\bar{x}_4$.

Метод Мак-Класски. Всем минитермам ставятся в соответствие их двоичные эквиваленты. Например, $x_1\bar{x}_2\bar{x}_3x_4 \Rightarrow 1001$. Пусть задана функция $f(X) = \bigcup(0, 1, 2, 3, 4, 6, 7, 8, 9, B, F)$. Далее все номера-минитермы разбиваются на группы по числу единиц в этих номерах:

0000 – нулевая группа;

0001, 0010, 0100, 1000 – первая группа;

0011, 0110, 1001 – вторая группа;

0111, 1011 – третья группа;

1111 – четвертая группа.

Осуществляется попарное склеивание только в группах-соседях. При формировании новых минитермов вместо исключенных переменных ставится прочерк:

000_, 00_0, 0_00, _000 – нулевая группа;

00_1, _001, 001_, 0_10, 01_0, 100_ – первая группа;

0_11, _011, 011_, 10_1 – вторая группа;

_111, 1_11 – третья группа.

Второй цикл склеиваний дает следующий результат:

00__, _00_, 0__0 – нулевая группа;

_0_1, 0_1_ – первая группа;

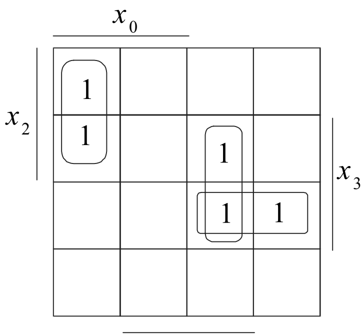
__11 – вторая группа.

Далее строится таблица импликант и расставляются метки, как и в методе Квайна:

	0000	0001	0010	0011	0100	0110	0111	1000	1001	1011	1111
00__	x	x	x	x							
_00_	x	x						x	x		
0__0	x		x		x	x					
_0_1		x		x					x	x	
0_1_			x	x		x	x				
__11				x			x			x	x

Результирующая МДНФ будет иметь вид $f(X) =$ $= \bar{x}_2\bar{x}_3 \cup \bar{x}_1\bar{x}_4 \cup x_3x_4$.

С использованием *карт Карно* можно достаточно просто минимизировать функции до четырех переменных. Карта Карно представляет собой таблицу, число клеток которой равно возможному числу наборов аргументов. В клетки, соответствующие минитермам СДНФ функции, вписываются единицы. В остальные клетки вписываются нули или клетки оставляются пустыми. Пусть задана функция $f(X) = \bigcup(5,8,10,13,14)$, для которой заполненная карта Карно будет иметь вид:



Далее соседние клетки с единицами объединяются в области. При объединении клеток следует придерживаться следующих правил: области должны быть прямоугольными с размерами сторон 1, 2 или 4 клетки; области должны быть как можно большего размера, областей должно быть как можно меньше; карту Карно следует рассматривать не как прямоугольник, а как развертку тора, т. е. соседними будут верхняя и нижняя строки, левый и правый столбцы; области могут пересекаться. Каждая область будет соответствовать простой импликанте МДНФ и состоять из тех аргументов, которые в этой области не меняют своего значения. Для данного примера МДНФ будет иметь вид $f(X) = x_0\bar{x}_1x_2 \cup \bar{x}_0x_1x_3 \cup \bar{x}_0\bar{x}_2x_3$.

КОМБИНАЦИОННЫЕ СХЕМЫ

Синтез комбинационных схем на интегральных микросхемах малой степени интеграции

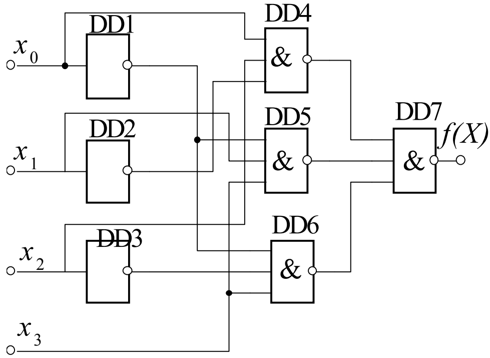
Комбинационной схемой (КС) называют совокупность элементов, реализующих заданную систему логических функций, значения которых однозначно определены значениями аргументов. Общим свойством КС является отсутствие петель, т. е. замкнутых цепей, по которым сигнал с выхода некоторого элемента непосредственно или через другие элементы может поступать на его вход.

Синтез производится в несколько этапов. На первом этапе следует получить МДНФ в форме, соответствующей выбранной элементной базе. Например, полученные ранее МДНФ для реализации на элементах штрих Шеффера следует преобразовать по правилу де Моргана в форму И–НЕ / И–НЕ:

$$f(X) = x_0\bar{x}_1x_2 \cup \bar{x}_0x_1x_3 \cup \bar{x}_0\bar{x}_2x_3 = \overline{(\overline{x_0\bar{x}_1x_2})(\overline{\bar{x}_0x_1x_3})(\overline{\bar{x}_0\bar{x}_2x_3})}.$$

На втором этапе при достаточном числе входов логических элементов строится схема, реализующая МДНФ:

Если число входов логических элементов недостаточно, переменные группируют в соответствии со свойствами ассоциативности логических операций дизъюнкции и конъюнкции.



Серийные логические элементы допускают определенное количество входов, подключаемых к одному выходу.

На третьем этапе устраняются перегрузки логических элементов с помощью дублирования – каждый перегруженный логический элемент заменяется двумя, тремя и так далее параллельно соединенными элементами с разделенными нагрузками. Особенное внимание следует уделять этому вопросу, если используются логические элементы различных серий. На этом этапе можно уменьшить число корпусов в КС. В одном корпусе интегральной микросхемы (ИМС) содержится несколько логических элементов, поэтому в некоторых случаях лишним логическим элементом с большим числом входов можно заменить элемент с меньшим числом входов.

Решение задачи синтеза КС со многими выходами можно свести к синтезу схемы с одним выходом. Для этого достаточно каждую из функций системы, описывающей схему, реализовывать отдельно. Однако при этом не учитывается то, что реализуемые функции зависят от одних и тех же аргументов. Вследствие этого их представления в виде МДНФ или МКНФ содержат одинаковые члены, реализация которых при таком подходе будет повторяющейся.

Методы синтеза схем со многими выходами, позволяющие избежать повторения элементов одинакового назначения, разделяют на две группы. К первой группе относятся методы, использующие нормальные (двухуровневые) канонические формы функций, ко второй – методы, использующие многоуровневое представление функций. Один из методов первой группы основан на совместной минимизации системы функций.

Простой импликантой системы функций $f_1(X), f_2(X), ..., f_n(X)$ называется такое элементарое произведение, которое является импликантой каждой из функций системы, но никакая его собственная часть не будет импликантой ни одной из функций. При совместной минимизации системы функций находят все простые импликанты каждой из функций, затем – простые импликанты всех возможных подсистем функций, состоящихиз двух функций, затем образуют подсистемы из трех функций и так далее.

На втором этапе минимизации из полученных простых импликант путем перебора различных вариантов отыскиваются наиболее простые формулы для представления функций системы. Отыскание таких формул удобно производить с помощью импликантных таблиц для системы функций, которые аналогичны соответствующим таблицам для одной функции.

Наиболее распространенный метод синтеза схем, относящийся ко второй группе, называется методом каскадов. В основе его лежит формула разложения функции

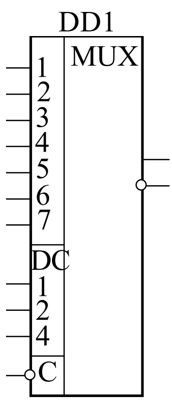
$$f(X) = x_1 f(1, x_2, x_3, ..., x_n) \cup \bar{x}_1 f(0, x_2, x_3, ..., x_n).$$

В свою очередь, к функциям $f(1, x_2, x_3,..., x_n)$ и $f(0, x_2, x_3,..., x_n)$ применяют эту же формулу до тех пор, пока в результате очередного разложения не будут получены переменные или их отрицания. Метод каскадов использует многоуровневые представления функций, поэтому задержка сигналов в схемах, синтезированных этим методом, больше. Выше рассмотрены частные случаи функции со многими выходами – полностью определенная функция. На практике функция может быть не полностью определена. В таких случаях следует придерживаться такого принципа: на наборах аргументов, где функция не определена, ее определяют так, чтобы соответствующая ей схема была наиболее простой.

Синтез комбынационных схем на интегральных микросхемах средней степени интеграции

Построение комбинационных устройств можно выполнять на основе мультиплексоров. *Мультиплексором* называется узел, выполняющий функции коммутации в одном направлении сигналов, приходящих из p возможных направлений.

Пример обозначения мультиплексора, для которого $p = 8$:



Этот мультиплексор содержит восемь информационных входов – $B_1, B_2,..., B_8$, три адресных входа – A_1, A_2, A_3 и один стробирующий (разрешающий) вход C. Логическая функция, реализуемая таким мультиплексором, имеет вид

$$y = \overline{C} B_1 \overline{A_1}\,\overline{A_2}\,\overline{A_3} \cup \overline{C} B_2 \overline{A_1}\,\overline{A_2} A_3 \cup \overline{C} B_3 \overline{A_1} A_2 \overline{A_3} \cup ...$$

$$... \cup \overline{C} B_8 A_1 A_2 A_3 .$$

Как видно, правая часть этого выражения представляет собой дизъюнкцию всех минитермов от переменных A_1, A_2, A_3. Таким образом, задавая на информационных входах мультиплексора константы 0 и 1, можно сформировать тот набор минитермов, который соответствует СДНФ заданной функции.

Однако *p*-входовой мультиплексор позволяет реализовывать не только функции n переменных, но и функции $n+1$ переменной. Пусть имеется функция $f(x_1, x_2, ..., x_n, x_{n+1})$. Введем следующие обозначения: $A_1 = x_1$, $A_2 = x_2$,..., $A_n = x_n$, $z = x_{n+1}$. Разложим исходную функцию в соответствии с *методом каскадов*:

$$f(A_1, A_2, ..., A_n, z) = \overline{A}_1 \overline{A}_2 ... \overline{A}_n f(0,0, ..., 0, z) \cup \overline{A}_1 \overline{A}_2 ...$$

$$... A_n f(0,0, ...,1, z) \cup ... \cup A_1 A_2 ... A_n f(1,1, ..., 1, z),$$

где функции $f(0,0, ..., 0, z), f(0,0, ...,1, z), ..., f(1,1, ..., 1, z)$ являются функциями одной переменной и, следовательно, могут принимать либо постоянные значения 0 и 1, либо значения z и $\bar{z}$. Сравнение полученного выражения с логической функцией, реализуемой мультиплексором, показывает, что логическую функцию $n + 1$ переменных можно реализовать на *p*-входовом мультиплексоре, заменив функции B_i на значения функции $f(i, z)$. Функции $f(i, z)$ можно определить непосредственно из таблицы истинности. Для этого достаточно сравнить значения $f(x_1, x_2, ..., x_n, x_{n+1})$ на соседних наборах $(i, 0)$ и $(i, 1)$, а затем воспользоваться следующими условиями:

если $f(i, 0) = 0$, $f(i, 1) = 0$, то $f(i, z) = 0$;

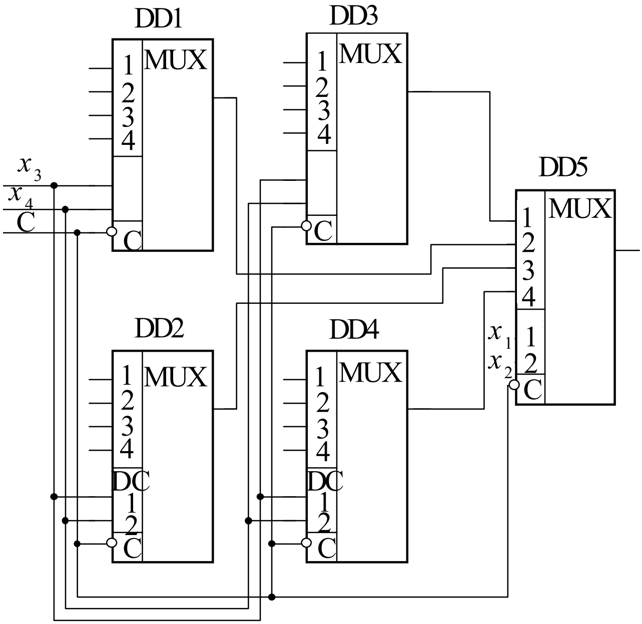
если $f(i, 0) = 0$, $f(i, 1) = 1$, то $f(i, z) = z$;

если $f(i, 0) = 1$, $f(i, 1) = 0$, то $f(i, z) = \overline{z}$;

если $f(i, 0) = 1$, $f(i, 1) = 1$, то $f(i, z) = 1$.

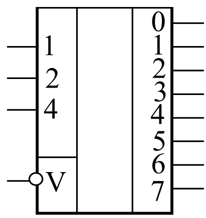
Последующая реализация производится аналогично случаю с функцией n переменных, но на информационные входы подаются не константы, а значения функции $f(i, z)$, т. е. 0, 1, z или $\bar{z}$.

Мультиплексор на требуемое число входов можно построить из мультиплексоров на меньшее число входов путем древовидного их включения:



Переменные x_1 и x_2 управляют подключением одного из мультиплексоров второго уровня к информационным входам мультиплексора первого уровня. Переменные x_3 и x_4 управляют коммутацией информационных входов мультиплексоров второго уровня. По аналогии можно построить мультиплексоры с большим числом входов и уровней.

Дешифратор (демультиплексор) также можно применять для реализации логических функций. Дешифратор можно представить как функциональный узел, коммутирующий входные сигналы V в одном из p возможных направлений. Условное графическое обозначение дешифратора для $p = 8$ имеет следующий вид:



Такой дешифратор формирует восемь функций:

$$D_0 = \overline{V}\,\overline{A}_1\overline{A}_2\overline{A}_3;\ D_1 = \overline{V}\,\overline{A}_1\overline{A}_2 A_3;$$

$$D_2 = \overline{V}\,\overline{A}_1 A_2\overline{A}_3;\ \ldots;\ D_7 = \overline{V} A_1 A_2 A_3.$$

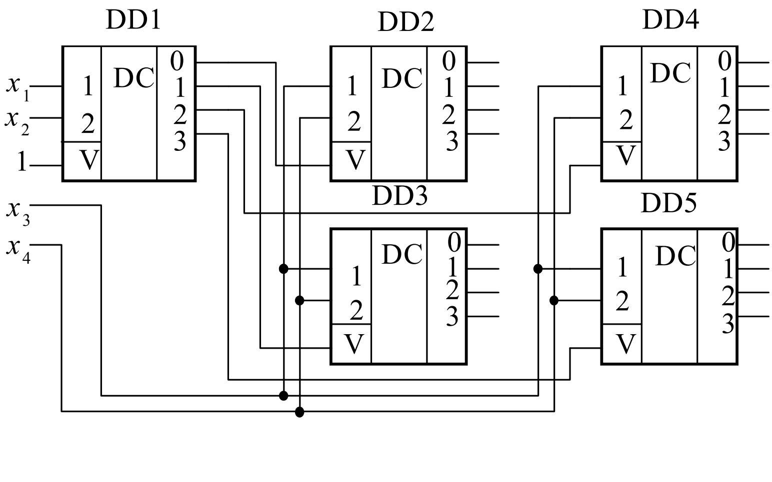
Легко заметить, что дешифратор формирует все возможные минитермы, которые можно образовать от n переменных. Следовательно, построение комбинационного устройства с использованием дешифратора сводится к объединению с помощью элемента ИЛИ тех выходов D_i, которые соответствуют минитермам заданной функции. При использовании мультиплексоров для каждой функции требуется отдельный мультиплексор. В случае применения дешифратора выходы одного из них можно использовать для реализации многих функций одних и тех же переменных.

Как следует из изложенного, ни один из минитермов не подвергается операции склеивания, даже если среди них есть соседние. Поэтому в тех случаях, когда в таблице истинности функции $f(X)$ нулей больше, чем единиц, для упрощения схемы целесообразно реализовывать инверсию функции $f(X)$, так как для функции $f(X)$ понадобятся элементы ИЛИ с меньшим числом входов. Это соответствует использованию при реализации вместо СДНФ канонической формы И–ИЛИ–НЕ. Большинство выпускаемых промышленностью дешифраторов имеют инверсные выходы, т. е. реализуют функции

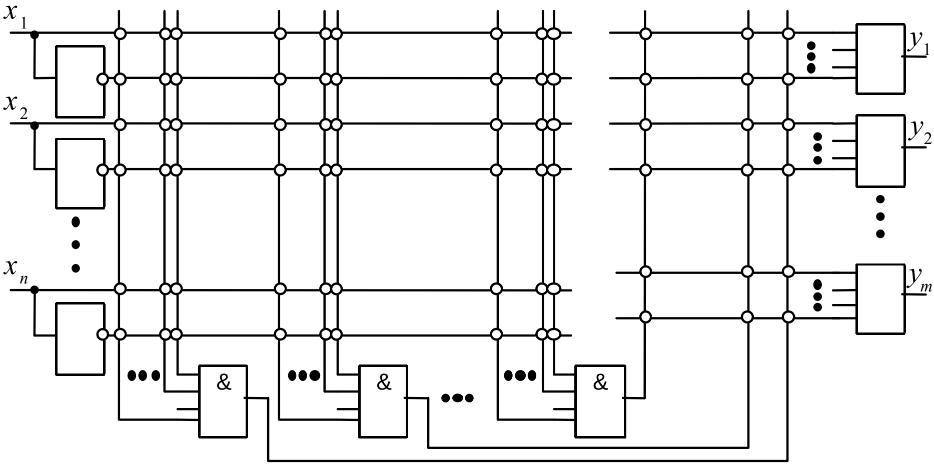
$$\overline{D}_0=\overline{\overline{V}\,\overline{A}_1\overline{A}_2\overline{A}_3};\ \overline{D}_1=\overline{\overline{V}\,\overline{A}_1\overline{A}_2A_3};\ \overline{D}_2=\overline{\overline{V}\,\overline{A}_1A_2\overline{A}_3};\ \cdots\ \overline{D}_7=\overline{\overline{V}A_1A_2A_3}.$$

При реализации логических функций с использованием таких дешифраторов для объединения выходов дешифратора следует использовать элементы И или И–НЕ. Это соответствует применению канонических форм И–НЕ / И или И–НЕ / И–НЕ.

Аналогично мультиплексорам можно наращивать разрядность дешифраторов по пирамидальному принципу. Например, дешифратор на четыре входа можно синтезировать с использованием пяти двухвходовых:



Для реализации сложных переключательных функций предпочтительно применение *программируемых логических матриц* (ПЛМ), представляющих собой ИМС с регулярной структурой, которые с помощью специальной процедуры (программирования) можно приспособить для реализации различных систем переключательных функций. Программирование ПЛМ сводится к устранению электрическим способом плавких перемычек в узлах И-матрицы и ИЛИ-матрицы. И-матрица содержит k n-входовых элементов И, каждый вход которых соединен с входными цепями с помощью плавких перемычек (показаны кружками). Это дает возможность сформировать в И-матрице k элементарных произведений с числом букв, не превышающим n. В свою очередь, ИЛИ-матрица содержит m k-входовых элементов ИЛИ, что позволяет формировать дизъюнкции не более чем по k различным произведениям, полученным с помощью И-матрицы.



Аналогичную структуру имеют *постоянные запоминающие устройства* (ПЗУ, ROM), но, в отличие от ПЛМ, имеющие возможность перестройки на уровне ИЛИ-матрицы. В отличие от случая использования мультиплексора или дешифратора при реализации комбинационной схемы на ПЛМ вопросы минимизации логической функции остаются актуальными. Параметры k, m, n вносят ограничения на число функций, реализуемых одной ПЛМ, число термов, входящих в ДНФ каждой функции, число переменных, входящих в каждый терм.

ЗАКЛЮЧЕНИЕ

В последние годы наметилось сокращение использования сложных комбинационных устройств на жесткой логике. Это связано с новыми подходами и направлениями развития электроники. В первую очередь, это экспансия на рынок встраиваемых систем управления однокристалльных ЭВМ и микроконтроллеров. Современные микроконтроллеры имеют, как правило, цену в несколько долларов США и способны реализовывать достаточно сложные алгоритмы, проигрывая жесткой логике только по быстродействию. Программируемые логические матрицы прекрасно справляются с задачей синтеза комбинационного устройства с сохранением высокого быстродействия, но в последнее время они вытесняются программируемыми вентильными матрицами, с помощью которых можно синтезировать не только комбинационные, но и конечные автоматы. Поэтому методы синтеза, изложенные выше, следует рассматривать не только с точки зрения синтеза устройств на жесткой логике, но и с точки зрения синтеза комбинационного автомата на любой другой элементной базе: микроконтроллера, программируемой вентильной матрицы, постоянного запоминающего устройства.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. *Титце У.*, *Шенк К.* Полупроводниковая схемотехника: Справочное руководство: Пер. с нем. – М.: Мир, 1982. – 512 с.

2. *Хоровиц П.*, *Хилл У.* Искусство схемотехники. – 5-е изд., перераб. и доп. – М.: Мир, 1998. – 704 с.

ОГЛАВЛЕНИЕ

Учебное издание

Дмитрий Валериевич КОСТЕНКО

ПРОЕКТИРОВАНИЕ КОМБИНАЦИОННЫХ УСТРОЙСТВ

Методические указания

(*на русском языке*)

Редактор Н.А. Шайкина
Компьютерная правка и верстка Е.А. Докиенко
Корректор Н.А. Шайкина

Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції ДК № 1150 від 12.12.2002 р.

Підписано до друку 16.05.06. Папір офсетний. Формат 60×84/16.
Гарнітура "Таймс". Друк офсетний. Ум. друк. арк. 1,4. Обл.-вид. арк. 1,5.
Тираж 100 прим. Вид. № 5. Зам. № 138. Ціна договірна

Видавець і виготівник Національний університет кораблебудування,
54002, м. Миколаїв, вул. Скороходова, 5

ДЛЯ НОТАТОК